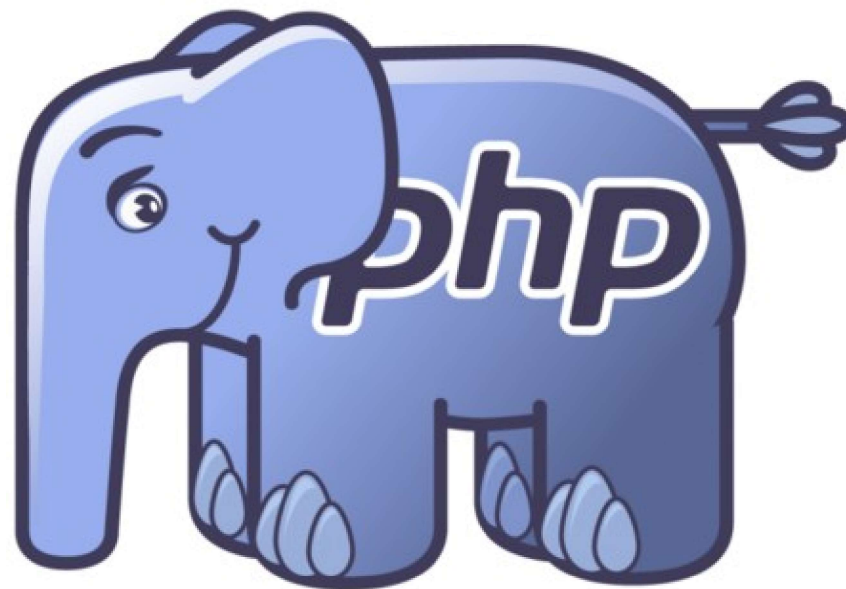


Métodos GET y POST



Métodos GET y POST

- ❑ Los métodos HTTP GET y HTTP POST permiten enviar información al servidor
- ❑ PHP administra esta información mediante los arrays:
 - **\$_GET**
 - **\$_POST**

Métodos GET y POST

□ Existen dos métodos con los que el navegador puede enviar información al servidor:

– Método HTTP GET.

- Información se envía de forma **VISIBLE**

– Método HTTP POST.

- Información se envía de forma **NO VISIBLE**

Métodos GET y POST

- ❑ Esta codificación es una cadena conformada por claves y valores separados por un ampersand (&)
 - clave1=valor1&clave2=valor2&clave3=valor3...
- ❑ Antes de que el navegador envíe la información proporcionada, la codifica mediante URL encoding, dando como resultado un Query String.
- ❑ Los espacios y otros caracteres no alfanuméricos se sustituyen.
- ❑ Una vez que la información es codificada, se envía al servidor.

Método HTTP GET

- ❑ El método GET envía la información codificada del usuario en el encabezado de la solicitud del HTTP, directamente en la URL.
- ❑ La página web y la información codificada se separan por un símbolo de interrogación (?):
 - `www.ejemplo.com/index.php?clave1=valor1&clave2=valor2`
 - `www.ejemplo.com/index.html?clave1=valor1&clave2=valor2`

Método HTTP GET

- ❑ El método **GET** envía la información en la propia URL, estando limitada a 2000 caracteres.
- ❑ La información es visible por lo que con este método nunca se envía información sensible.
- ❑ No se pueden enviar datos binarios (archivos, imágenes...).
- ❑ En PHP los datos se administran con el array `$_GET`.

Método HTTP GET

- Un ejemplo básico de un formulario html con el método GET:

```
<html>
<body>
  <form action="ejemplo_form_get.php" method="get">
    Nombre: <input type="text" name="nombre"><br>
    Email: <input type="text" name="email"><br>
    <input type="submit" value="Enviar">
  </form>
  Hola <?php
    if (isset($_GET["nombre"]))
      echo $_GET["nombre"] . ":" ;
  ?><br>
  Tu email es: <?php
    if (isset($_GET["email"]))
      echo $_GET["email"] . ":";
  ?>
</body>
</html>
```

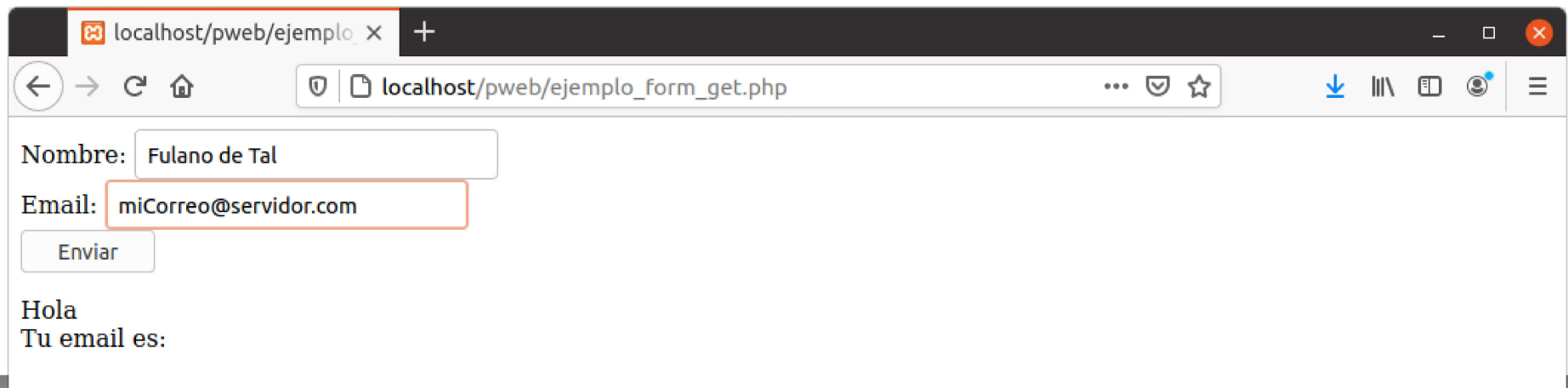
Método HTTP GET

□ Probando el ejemplo anterior



A screenshot of a web browser window. The address bar shows the URL `localhost/pweb/ejemplo_form_get.php`. The page content includes a form with two input fields: "Nombre:" and "Email:". Below the "Email:" field is a button labeled "Enviar". Underneath the form, the text "Hola" and "Tu email es:" is visible.

□ Ingresamos datos de prueba y presionamos el botón “Enviar”



A screenshot of the same web browser window, but now the form fields are filled with test data. The "Nombre:" field contains the text "Fulano de Tal" and the "Email:" field contains "miCorreo@servidor.com". The "Enviar" button is still present. The text "Hola" and "Tu email es:" remains at the bottom of the page.

Método HTTP GET

- ❑ La URL que resulta al hacer clic en el botón “enviar” es de la forma:

**ejemplo_form_get.php?nombre=Fulano+de+Tal&
email=miCorreo%40servidor.com**

- En este caso @ es un carácter especial y se codifica (%40).
- Los espacios también se codifican (+)

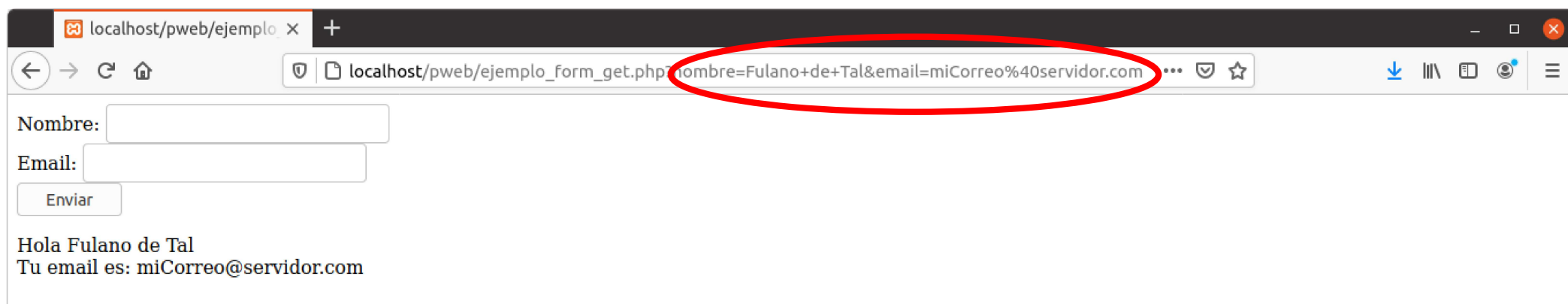


Método HTTP GET

□ La URL que resulta al hacer clic en el botón “enviar” es de la forma:

– `ejemplo_form_get.php?nombre=Fulano+de+Tal&email=miCorreo%40servidor.com`

- En este caso @ es un carácter especial y se codifica (%40).
- Los espacios también se codifican (+)



Método HTTP POST

- ❑ Con el método HTTP POST también se codifica la información, pero ésta se envía a través del cuerpo de la solicitud HTTP, por lo que no aparece en la URL.
- ❑ El método POST no tiene límite de cantidad de información a enviar.
- ❑ La información no es visible, por lo que se puede enviar información sensible.
- ❑ Se puede enviar texto así como datos binarios (archivos, imágenes...).
- ❑ PHP proporciona el array `$_POST` para acceder a la información enviada.

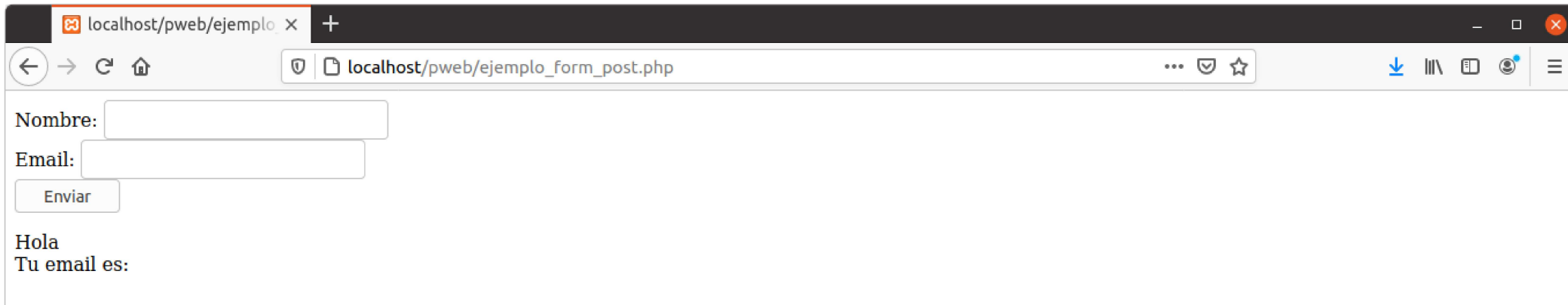
Método HTTP POST

- Un ejemplo básico de un formulario html con el método POST:

```
<html>
<body>
  <form action="ejemplo_form_post.php" method="post">
    Nombre: <input type="text" name="nombre"><br>
    Email: <input type="text" name="email"><br>
    <input type="submit" value="Enviar">
  </form>
  Hola <?php
    if (isset($_POST["nombre"]))
      echo $_POST["nombre"] . ":" ;
  ?><br>
  Tu email es: <?php
    if (isset($_POST["email"]))
      echo $_POST["email"] . ":";
  ?>
</body>
</html>
```

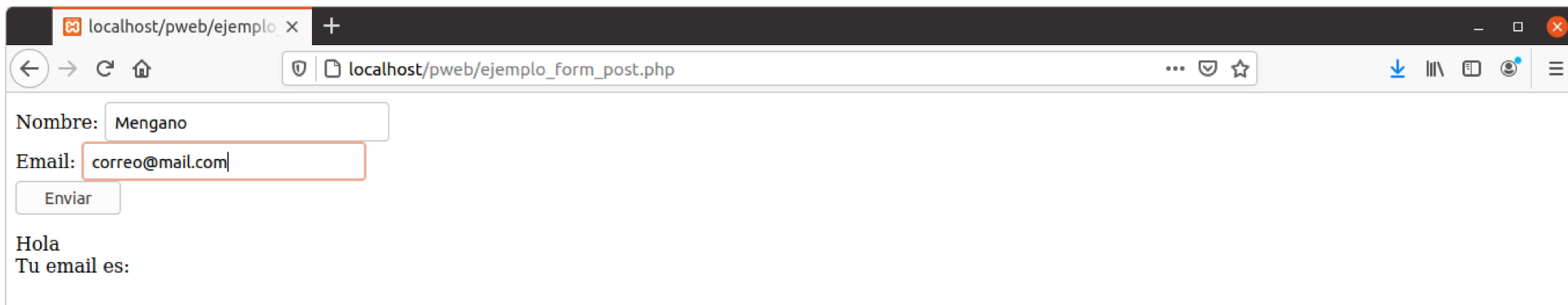
Método HTTP POST

□ Probando el ejemplo anterior



A screenshot of a web browser window. The address bar shows the URL `localhost/pweb/ejemplo_form_post.php`. The page content includes a form with two input fields: "Nombre:" and "Email:". Below the "Email:" field is a button labeled "Enviar". Underneath the form, the text "Hola" and "Tu email es:" is displayed.

□ Ingresamos datos de prueba y presionamos el botón “Enviar”



A screenshot of a web browser window, similar to the previous one, but with test data entered. The "Nombre:" field contains the text "Mengano" and the "Email:" field contains "correo@mail.com". The "Enviar" button is highlighted with a red border, indicating it has been clicked or is the focus of the action.

Método HTTP POST

- ❑ Como se puede comprobar la información no se muestra en la URL.

